**Linux 指令：**

**ls:**

-a, –all 列出目录下的所有文件，包括以 . 开头的隐含文件

-g 类似 -l,但不列出所有者

-G, –no-group 不列出任何有关组的信息

-h, –human-readable 以容易理解的格式列出文件大小 (例如 1K 234M 2G)

–si 类似 -h,但文件大小取 1000 的次方而不是 1024

-i, –inode 印出每个文件的 inode 号

-I, –ignore=样式 不印出任何符合 shell 万用字符<样式>的项目

-k 即 –block-size=1K,以 k 字节的形式表示文件的大小

-R, –recursive 同时列出所有子目录层

-t 以文件修改时间排序

-**列出当前目录中所有以“t”开头的目录的详细内容，可以使用如下命令：**

**命令**：ls -l t\*

列出 /opt/soft 文件下面的子目录:

**命令：**ls -F /opt/soft |grep /$

**例四**：列出目前工作目录下所有名称是s 开头的档案，愈新的排愈后面，可以使用如下命令：

**命令**：ls -ltr s\*

**例十：递归列出当前目录下的所有文件（包括隐藏文件）的绝对路径**

**命令**： find $PWD | xargs ls -ld

**pwd:**

一般情况下不带任何参数，如果目录是链接时，则：

**格式：**pwd -P  显示出实际路径，而非使用连接（link）路径

**命令：**

/bin/pwd [选项]

选项：

-L 目录连接链接时，输出连接路径

-P 输出物理路径

**cd:**

**返回进入此目录之前所在的目录 cd-**

**把上个命令的参数作为cd参数使用 cd!$**

**mkdir:**

用来创建指定的名称的目录，要求创建目录的用户在当前目录中具有**写权限**，并且指定的目录名**不能**是当前目录中已有的目录.

mkdir [选项] 目录...

-m, --mode=模式，设定权限<模式> (类似 chmod)，而不是 rwxrwxrwx 减 umask

-p, --parents  可以是一个路径名称。此时若路径中的某些目录尚不存在,加上此选项后,系统将自动建立好那些尚不存在的目录,即一次可以建立多个目录;

-v, --verbose  每次创建新目录都显示信息

--help   显示此帮助信息并退出

--version  输出版本信息并退出

实例3：创建权限为777的目录

mkdir -m 777 test3

实例2：递归创建多个目录

mkdir -p test2/test22

实例五：一个命令创建项目的目录结构

mkdir -vp scf/{lib/,bin/,doc/{info,product},logs/{info,product},service/deploy/{info,product}}

**rm:**

rm是常用的命令，该命令的功能为删除一个目录中的一个或多个文件或目

录，它也可以将某个目录及其下的所有文件及子目录均删除。对于链接文件，只是删除了链接，原有文件均保持不变

rm [选项] 文件…

-f, --force    忽略不存在的文件，从不给出提示。

-i, --interactive 进行交互式删除

-r, -R, --recursive   指示rm将参数中列出的全部目录和子目录均递归地删除。-v, --verbose    详细显示进行的步骤

       --help     显示此帮助信息并退出

       --version  输出版本信息并退出

输入rm log.log命令后，系统会询问是否删除，输入y后就会删除文件，不想删除则是n

**实例六：**删除以 -f 开头的文件

rm -- -f

**rmdir:**

rmdir是常用的命令，该命令的功能是删除空目录，**一个目录被删除之前必须是空的**。（注意，rm - r dir命令可代替rmdir，但是有很大危险性。）删除某目录时也必须具有对父目录的写权限。

**实例2：**rmdir -p 当子目录被删除后使它也成为空目录的话，则顺便一并删除

rmdir -p logs

**mv:**

mv命令是move的缩写，可以用来移动文件或者将文件改名（move (rename) files），是Linux系统下常用的命令，经常用来备份文件或者目录。

mv [选项] 源文件或目录 目标文件或目录

-b ：若需覆盖文件，则覆盖前先行备份。

-f ：force 强制的意思，如果目标文件已经存在，不会询问而直接覆盖；

-i ：若目标文件 (destination) 已经存在时，就会询问是否覆盖！

-u ：若目标文件已经存在，且 source 比较新，才会更新(update)

-t  ： --target-directory=DIRECTORY move all SOURCE arguments into DIRECTORY，即指定mv的目标目录，该选项适用于移动多个源文件到一个目录的情况，此时目标目录在前，源文件在后。

**实例一：文件改名**

mv test.log test1.txt 将文件test.log重命名为test1.txt

**实例二：移动文件**

mv test1.txt test3 将test1.txt文件移到目录test3中

**实例三**：将文件log1.txt,log2.txt,log3.txt移动到目录test3中。

mv log1.txt log2.txt log3.txt test3

mv -t /opt/soft/test/test4/ log1.txt log2.txt  log3.txt

mv log1.txt log2.txt log3.txt test3 命令将log1.txt ，log2.txt， log3.txt 三个文件移到 test3目录中去.

mv -t /opt/soft/test/test4/ log1.txt log2.txt log3.txt 命令又将三个文件移动到test4.

**实例四：将文件file1改名为file2，如果file2已经存在，则询问是否覆盖**

mv -i log1.txt log2.txt

**实例六：目录的移动**

mv dir1 dir2 如果目录dir2不存在，将目录dir1改名为dir2；否则，将dir1移动到dir2中

**实例八：把当前目录的一个子目录里的文件移动到另一个子目录里**

mv test3/\*.txt test5

**cp:**

cp命令用来复制文件或者目录，是Linux系统中最常用的命令之一.

将源文件复制至目标文件，或将多个源文件复制至目标目录:

cp [选项]... [-T] 源 目的

或：cp [选项]... 源... 目录

或：cp [选项]... -t 目录 源...

-a, --archive , 为每个已存在的目标文件创建备份

-b  类似--backup 但不接受参数, --copy-contents        在递归处理是复制特殊文件内容

-d  等于--no-dereference --preserve=links

f, --for如果目标文件无法打开则将其移除并重试(当-n选项存在时则不需再选此项)

-i, --interactive  覆盖前询问(使前面的 -n 选项失效)

-H  跟随源文件中的命令行符号链接

-P, --no-dereference   不跟随源文件中的符号链接

-l, --link  链接文件而不复制

-L, --dereference  总是跟随符号链接

-n, --no-clobber  不要覆盖已存在的文件(使前面的 -i 选项失效)

-R, -r, --recursive  复制目录及目录内的所有项目

**touch:**

linux的touch命令不常用，一般在使用make的时候可能会用到，用来修改文件时间戳，或者新建一个不存在的文件。

touch [选项]... 文件..

-a   或--time=atime或--time=access或--time=use 　只更改存取时间。

-c   或--no-create 　不建立任何文档。

-d 　使用指定的日期时间，而非现在的时间。

-f 　此参数将忽略不予处理，仅负责解决BSD版本touch指令的兼容性问题。

-m   或--time=mtime或--time=modify 　只更改变动时间。

-r 　把指定文档或目录的日期时间，统统设成和参考文档或目录的日期时间相同。

-t 　使用指定的日期时间，而非现在的时间

**实例一：创建不存在的文件**

如果log2014.log不存在，则不创建文件

[root@localhost test]# touch -c log2014.log

[root@localhost test]# ll

-rw-r--r-- 1 root root    0 10-28 16:01 log2012.log

-rw-r--r-- 1 root root    0 10-28 16:01 log2013.log

**实例二：更新log.log的时间和log2012.log时间戳相同**

[root@localhost test]# ll

-rw-r--r-- 1 root root    0 10-28 16:01 log2012.log

-rw-r--r-- 1 root root    0 10-28 16:01 log2013.log

-rw-r--r-- 1 root root    0 10-28 14:48 log.log

[root@localhost test]# touch -r log.log log2012.log

[root@localhost test]# ll

-rw-r--r-- 1 root root    0 10-28 14:48 log2012.log

-rw-r--r-- 1 root root    0 10-28 16:01 log2013.log

-rw-r--r-- 1 root root    0 10-28 14:48 log.log

**cat:**

cat命令的用途是连接文件或标准输入并打印。这个命令常用来显示文件内容，或者将几个文件连接起来显示，或者从标准输入读取内容并显示，它常与重定向符号配合使用

-A, --show-all           等价于 -vET

-b, --number-nonblank    对非空输出行编号

-e                       等价于 -vE

-E, --show-ends          在每行结束处显示 $

-n, --number     对输出的所有行编号,由1开始对所有输出的行数编号

-s, --squeeze-blank  有连续两行以上的空白行，就代换为一行的空白行

-t                       与 -vT 等价

-T, --show-tabs          将跳格字符显示为 ^I

-u                       (被忽略)

-v, --show-nonprinting   使用 ^ 和 M- 引用，除了 LFD 和 TAB 之外

**nl:**

nl命令在[linux](http://codex.wordpress.org.cn/Linux)系统中用来计算文件中行号。nl 可以将输出的文件内容自动的加上行号！其默认的结果与 cat -n 有点不太一样， nl 可以将行号做比较多的显示设计与是否自动补齐 0 等等的功能。

nl [选项]... [文件]...

-b  ：指定行号指定的方式，主要有两种：

-b a ：表示不论是否为空行，也同样列出行号(类似 cat -n)；

-b t ：如果有空行，空的那一行不要列出行号(默认值)；

-n  ：列出行号表示的方法，主要有三种：

-n ln ：行号在萤幕的最左方显示；

-n rn ：行号在自己栏位的最右方显示，且不加 0 ；

-n rz ：行号在自己栏位的最右方显示，且加 0 ；

-w  ：行号栏位的占用的位数。

-p 在逻辑定界符处不重新开始计算

**more:**

more命令，功能类似 cat ，cat命令整个文件的内容从上到下显示在屏幕上。 more会以一页一页的显示方便使用者逐页阅读，而最基本的指令就是按空白键（space）就往下一页显示，按 b 键就会往回（back）一页显示，而且还有搜寻字串的功能 。more命令从前向后读取文件，因此在启动时就加载整个文件。

**1．命令格式：**

more [-dlfpcsu ] [-num ] [+/ pattern] [+ linenum] [file ... ]

**2．命令功能：**

more命令和cat的功能一样都是查看文件里的内容，但有所不同的是more可以按页来查看文件的内容，还支持直接跳转行等功能。

**3．命令参数：**

+n      从笫n行开始显示

-n       定义屏幕大小为n行

+/pattern 在每个档案显示前搜寻该字串（pattern），然后从该字串前两行之后开始显示

-c       从顶部清屏，然后显示

-d       提示“Press space to continue，’q’ to quit（按空格键继续，按q键退出）”，禁用响铃功能

-l        忽略Ctrl+l（换页）字符

-p       通过清除窗口而不是滚屏来对文件进行换页，与-c选项相似

-s       把连续的多个空行显示为一行

-u       把文件内容中的下画线去掉

**4．常用操作命令：**

Enter    向下n行，需要定义。默认为1行

Ctrl+F   向下滚动一屏

空格键  向下滚动一屏

Ctrl+B  返回上一屏

=       输出当前行的行号

：f     输出文件名和当前行的行号

V      调用vi编辑器

!命令   调用Shell，并执行命令

q       退出more

**实例4：列一个目录下的文件，由于内容太多，我们应该学会用more来分页显示。这得和管道 | 结合起来**

**命令：**

ls -l  | more -5

**less:**

less 工具也是对文件或其它输出进行分页显示的工具，应该说是linux正统查看文件内容的工具，功能极其强大。less 的用法比起 more 更加的有弹性。在 more 的时候，我们并没有办法向前面翻， 只能往后面看，但若使用了 less 时，就可以使用 [pageup] [pagedown] 等按键的功能来往前往后翻看文件，更容易用来查看一个文件的内容！除此之外，在 less 里头可以拥有更多的搜索功能，不止可以向下搜，也可以向上搜

**head & tail :**

head 与 tail 就像它的名字一样的浅显易懂，它是用来显示开头或结尾某个数量的文字区块，head 用来显示档案的开头至标准输出中，而 tail 想当然尔就是看档案的结尾

**chown $ chmod:**

$chmod 755 a.txt

$sudo chown root a.txt

**Linux 文件名通配表达式 :**

有时候，我们想列出工作目录下所有的以.txt结尾的文件的信息，可以用下面的方式:

$ls -l \*.txt

\*.txt的写法就运用了Linux通配表达式。它与正则表达式相类似，但语法有所不同。

Filename Pattern Matching       对应含义

\*                               任意多个任意字符

?                               任意**一个**字符

[kl]                            字符k或者字符l

[0-4]                           数字0到4字符中的一个

[b-e]                           b到e字符中的一个

[^mnp]                          一个字符，这个字符不是m,n,p

**注意：**

$rm \* .txt

$rm \*.txt

第一个命令会删除当前目录下所有文件

**Linux 重新定向输入或输出：**

**定向输出：**

$ls > a.txt

重新定向标准输出。这里的>就是提醒命令行，让它知道我现在想变换文本流的方向了，我们不让标准输出输出到屏幕，而是要到a.txt这个文件 (好像火车轨道换轨)。此时，计算机会新建一个a.txt 的文件，并将命令行的标准输出指向这个文件。

$ls >> a.txt

这里>>的作用也是重新定向标准输出。如果a.txt已经存在的话，ls产生的文本流会附加在a.txt的结尾，而不会像>那样每次都新建a.txt。

$echo IamVamei

echo的作用是将文本流导向标准输出。在这里，echo的作用就是将IamVamei输出到屏幕上。如果是

$echo IamVamei > a.txt

a.txt中就会有IamVamei这个文本。

**定向输入：**

我们也可以用<符号来改变标准输入。比如cat命令，它可以从标准输入读入文本流，并输出到标准输出:

$cat < a.txt

我们将cat标准输入指向a.txt，文本会从文件流到cat，然后再输出到屏幕上。当然，我们还可以同时重新定向标准输出:

$cat < a.txt > b.txt

这样，a.txt的内容就复制到了b.txt中。

**定向标准输出和标准错误：**

我们还可以使用>&来同时重新定向标准输出和标准错误。假设我们并没有一个目录void。那么

$cd void > a.txt

会在屏幕上返回错误信息。因为此时标准错误依然指向屏幕。当我们使用:

$cd void >& a.txt

错误信息被导向a.txt。

如果只想重新定向标准错误，可以使用2>:

$cd void 2> a.txt > b.txt

标准错误对应的总是2号，所以有以上写法。标准错误输出到a.txt，标准输出输出到b.txt。

例如：

$cat < a.txt | wc

wc命令代表word count，**用于统计文本中的行、词以及字符的总数**。a.txt中的文本先流到cat，然后从cat的标准输出流到wc的标准输入，从而让wc知道自己要处理的是a.txt这个字符串。

**alias $ unalias:**

alias 要建的别名的对象＝‘原名字‘(注意有单引号)

alias 列出已经建立的别名列表

unalias 取消别名。

**输入$输出重定向:**

**输出：> 将一个命令输出放入文件且覆盖原文件**

**>> 不会覆盖原文件，在原文件末尾追加**

**输入： <**

**chmod:**

**1）. 文字设定法:**　　 **chmod ［who］ ［+ | - | =］ ［mode］ 文件名**

**2）. 数字设定法**

r=4，w=2，x=1  
若要rwx属性则4+2+1=7  
若要rw-属性则4+2=6；  
若要r-x属性则4+1=7。

**权限范围：**  
u ：目录或者文件的当前的用户  
g ：目录或者文件的当前的群组  
o ：除了目录或者文件的当前用户或群组之外的用户或者群组  
a ：所有的用户及群组

类似的有

chgrp [选项] [组] [文件]

变更文件与目录所属群组

chown

将指定文件的拥有者改为指定的用户或组，用户可以是用户名或者用户ID；组可以是组名或者组ID；文件是以空格分开的要改变权限的文件列表，支持通配符

chown [选项]... [所有者][:[组]] 文件...

关于目录的权限定义：

|  |  |  |
| --- | --- | --- |
| 文件类型 | 目录 | 文件 |
| 读 | 可以查看目录内容，如ls命令就需要目录有读权限 | 可以打开文件查看其中内容，如cat,more,head,tail |
| 写 | 可以再该目录下添加，删除子目录和文件，如rm,rmdir,mkdir命令需要目录有写权限 | 可以打开文件并且可以修改文件内容，如vi,cp |
| 执行 | 可以进入到该目录，如cd目录 | 可以执行文件 |

文件查找：

find path expression [action]

例如：find . mtime -2 查找当前目录下两天前修改的文件

find /home –amin -10 查找10**分钟**内存取的文件或目录

find /home –atime -10 查找10**小时**内存取的文件或目录

locate [-d <数据库文件>] ［--help］[--version][keywords]

文件过滤和统计：

grep: 查找包含某些字符串的结果，对文件或输出结果进行过滤，大小写敏感。

grep [option(s)] string filename

例如：

ls -la | grep -i 'nov 2' (找出修改日期是nov 2 的文件)

wc:统计文件或结果

wc [options(s)] filename(s)

**在 etc/inittab 的id:5:initdefault**

0:关机 3:多用户状态有网 5:图形界面 6:系统重启

etc/group 文件里面存有用户群组信息。

添加一个组：groupadd policeman

etc/passwd 文件里面有用户信息。

添加一个用户并且指定组：useradd –g 组名 用户名

etc/profile 配置环境变量文件

tar –zxvf 文件名.tar.gz 安装此文件

./eclipse 启动文件 后缀为bin也是此方法打开

**mnt下linux 分区的几个重要概念：**

mount [-参数] ［设备名称］ ［挂在点］ =>挂载

（特别的，挂载光驱可以用mount /mnt/cdrom）

umount [设备名称] =>卸载

df [-参数] =>查看磁盘使用情况

df [目录全路径]=> 查看目录是哪个分区

fdisk –l ＝>查看linux 系统下分区具体情况

ln –s 创建的文件 原文件 =>创建一个链接文件

**etc/profile:**

**配置系统的环境变量**

**/home/TianRan/.bash\_profile :**

**配置用户的环境变量**

.bashrc**:用户登录的时候就执行,因此配置此文件可以指定某些程序在用户登录的时候就自动启动。注意每个用户下都有一套此相应的文件，修改etc/.bashrc 的 文件是对所有的用户都有效。**

**Export 临时添加环境变量：**

**export PATH=$PATH:我们要加的路径名字(比如/root,就添加了一个/root目录到环境变量了，可以通过env查看环境变量)**

**压缩和解压缩文件**

**zip aa.zip 文件1 文件2 =>压缩两个文件**

**zip –r aa.zip 文件夹路径 =>压缩整个路径下的所有文件**

**unzip aa.zip**

**用gzip 和 gunzip处理.gz 文件。**